CLAIMS

1. A real-time video decoder for use with a mobile device, comprising:

means for receiving a system stream, said system stream comprising:

a system layer containing timing and other information needed to demultiplex audio and video streams and to synchronize audio and video during playback; and

a compression layer comprising said audio and video streams;

a system decoder for extracting timing information from a system stream and sending said timing information to a other system components, said system decoder also demultiplexing said video and audio streams from said system stream and then sending each of said video and audio streams to a corresponding decoder;

a video decoder for decompressing said video stream; and

an audio decoder for decompressing said audio stream.

2. The decoder of Claim 1, wherein the MP3 audio compression standard is used as a default audio format.

3. The decoder of Claim 1, further comprising:

an encryption facility comprising an encryption algorithm based on the Blowfish algorithm.

4. In a decoding technique, an audio/video (AV) synchronization method, comprising the steps of:

assigning each decompressed video frame in a video stream a unique id (0,1,2,3,...);

assigning each audio packet in an audio stream a unique id (0,1,2,3...);

using an AV sync code to monitor the ids of a latest rendered video frame and audio packet;

recalculating said ids into real time stamps very time a video interrupt occurs; and

using said AV sync code to compare said time stamps and determine whether a next video frame must be repeated or dropped;

wherein said audio stream is never adjusted; and

wherein video frames are either skipped or repeated to fit a current audio position.

5. A method for reducing the complexity of MPEG4 decoding, comprising the steps of:

disabling intra prediction of AC coefficients, wherein a flag that indicates the need for AC prediction is eliminated from a MPEG4 bitstream;

disabling motion compensation rounding control, wherein a rounding bit is eliminated from said MPEG4 bitstream;

combining VLC decoding and dequantization into one step, wherein dequantization of a coefficient is made immediately after decoding its variable length code, and wherein zero coefficients are exclude from dequantization; and

simplifying inverse discrete cosine transformation with a significance map, wherein said significance map stores positions of last nonzero coefficients in each row/column of a discrete cosine transformation block, wherein said significance map is filled during VLC decoding.

6. The method of Claim 5, wherein two different versions of inverse discrete cosine transformation are provided: one for rows/columns of eight coefficients and one for rows/columns of three coefficients, wherein if all coefficients in row/column are zero coefficients, inverse transformation is not performed.

7. A method for reducing the complexity of MPEG4 decoding, comprising the step of:

disabling intra prediction of AC coefficients, wherein a flag that indicates the need for AC prediction is eliminated from a MPEG4 bitstream.

8. A method for reducing the complexity of MPEG4 decoding, comprising the step of:

disabling motion compensation rounding control, wherein a rounding bit is eliminated from said MPEG4 bitstream.

9. A method for reducing the complexity of MPEG4 decoding, comprising the step of:

combining VLC decoding and dequantization into one step, wherein dequantization of a coefficient is made immediately after decoding its variable length code, and wherein zero coefficients are exclude from dequantization.

10. A method for reducing the complexity of MPEG4 decoding, comprising the step of:

simplifying inverse discrete cosine transformation with a significance map, wherein said significance map stores positions of last nonzero coefficients in each row/column of a discrete cosine transformation block, wherein said significance map is filled during VLC decoding.

11. The method of Claim 10, wherein two different versions of inverse discrete cosine transformation are provided: one for rows/columns of eight coefficients and one for rows/columns of three coefficients, wherein if all coefficients in row/column are zero coefficients, inverse transformation is not performed.

12. A method for fast "YUV to RGB555" conversion, comprising the steps of:

providing a conversion table; and

calculating a table index as a function of three colors in YUV format;

wherein a conversion table cell represents a color in RGB555 format that corresponds to a color in YUV format.

13. The method of Claim 12, wherein YUV format is represented as:

```
Index = ((U >> (8-BITS_U)) << (BITS_Y+BITS_V)) +  ((V
>> (8-BITS_V)) << (                BITS_V)) + (Y >> (8-
BITS_Y))
```

where Y, U, and V are 8-bit color components in YUV format; and BITS_Y, BITS_U, BITS_V are the numbers of significant bits for each color: Y, U, and V.

14. The method of Claim 13, wherein the number of indexes is (1 << (BITS_Y+BITS_U+BITS_V)), wherein the size of a cell is two bytes (high-order bit is unused), and wherein the size of said table is the number of indexes * 2, that is:

```
(1 << (BITS_Y+BITS_U+_BITS_V + 1)).
```

15. The method of Claim 14, wherein the number of significant bits for the Y color component must be greater than number of significant bits for the U and V components.

16. The method of Claim 15, wherein said color conversion table is organized to avoid cache misses during conversion of image in YUV 4:2:0 format.

17. A method for compressing b-frames, comprising the steps of:

storing several video frames in one chunk; and

inserting large amounts of empty (zero length) video chunks into an AV stream to isolate audio chunks.

18. The method of Claim 17, further comprising the step of:

applying a post-processing utility to an AVI file that isolates each video frame in its own chunk and drops all empty chunks.

19. A method for fast fixed-point implementation of an MPEG-1 Layer 3 decoding algorithm, comprising the steps of:

representing data as a sum of high and low parts:

```
y =
xHigh*cHigh+((xLow*cHigh+cLow*xHigh)>>12)+((xLow*cLow)>>24);
```

and

removing small parts from said sum.

20. The method of Claim 19, comprising a high precision summing step as follows:

```
y = xHigh*cHigh+((xLow*cHigh+cLow*xHigh)>>12).
```

21. The method of Claim 19, comprising a medium and low precision step as follows:

```
y = xHigh*cHigh + ((xLow*cHigh) >> 12).
```

22. The method of Claim 19, comprising a simplified multiplication on constant coefficients in 32.24 representation implemented as:

```
y = ((x>>6)*c)>>6,
```

in assumption that

$$| c_{float} | < 1;$$

wherein if

$$1.0 < | c_{float} | < 2.0,$$

said multiplication is performed as

$$y = ((x>>6)*c)>>5$$

where

$$c = (\text{int})(c_{\text{float}} *(1<<12)+0.5),$$

wherein if

$$1.0 < | c_{\text{float}} | < (1 << q),$$

using multiplication in a form:

$$y = ((x >> 6)*c) >> (6-q)$$

where

$$c = (\text{int})(c_{\text{float}}*(1<<(12-q))+0.5).$$

23. A method for computational speedup of an Inverse Modified Discrete Cosine Transform (IMDCT) calculation, comprising the step of:

using a simplified multiplication by transform coefficients.

24. The method of Claim 23, wherein for an IDMCT calculation on 36 and 12 points, transform coefficients with absolute values smaller than 1 are represented in 32.15 format, multiplication is by the coefficient:

$$y = ((x>>6)*c)>>6,$$

and, wherein for coefficients with absolute values greater than 1 multiplication is by the coefficient:

$$y = ((x >> 6)*c) >> (6-q).$$

25. The method of Claim 23, wherein for an IDMCT calculation on 64 points (synthesis function), where all transform coefficients have absolute value smaller than 1, and are represented in 32.15 format, multiplication is by the coefficient:

$$y = ((x>>6)*c)>>6.$$

26. The method of Claim 23, wherein for an IDMCT calculation in high precision mode,multiplication is by the coefficient:

$$y = xHigh*cHigh + ((xLow*cHigh) >> 12).$$

27. A method for computational speedup for a final windowing operation in an AV decoder, comprising the steps of:

calculating convolution of output samples by any of the following methods:

using a scaled transposed window table;

optimizing a convolution loop;

reducing a window table.

28. A multimedia file format for a compression/decompression facility, said file format holding highly compressed digital video, audio, graphics, and navigation data, said file format comprising:

a main configuration file for multimedia file storage media that specifies the media, a main navigation script file name, and a decoding engine to use;

multiplexed compressed video/audio streams;

menu subpictures comprising compressed bitmaps;

navigation scripts for video chapters which specify an order in which chapters are played; and

menu files that describe menu representation and functionality by specifying subpictures for menu items, pointers to chapters, and the like.

29. The file format of Claim 28, wherein multimedia encoded in said multimedia file format is protected from unauthorized copying using a highly secure encryption scheme based on the Blowfish algorithm.

30. The file format of Claim 29, wherein a plurality of different keys are generated using a strong random number generator, wherein said keys are scrambled, and wherein said keys stored at various offsets within a system internal memory.

31. The file format of Claim 30, wherein different keys are used to encrypt prerecorded content, downloaded content, and code updates.

32. A multimedia encryption method for a portable device, comprising the steps of:

storing prerecorded content on an SD or MMC memory card which contains a unique card key which is stored in a protected area of said card;

storing a player key within a portable device internal memory, wherein said player key is modified by said unique card key to produce a new key; and

encrypting said content with said new key prior to storing said content in said memory card;

wherein multimedia encoded in said multimedia file format on said memory card is protected from unauthorized copying using a highly secure encryption algorithm; and

wherein content cannot be copied onto another memory card and played back without knowledge of said player key, said card key, and said encryption algorithm.

33. A multimedia encryption method for a portable device, comprising the steps of:

encrypting downloaded content with a separate player key, wherein said player key is modified by a unique player ID;

uploading said player ID to a content server prior to downloading content;

after downloading, copying said content onto an SD or MMC memory card;

wherein multimedia encoded in said multimedia file format on said memory card is protected from unauthorized copying using a highly secure encryption algorithm; and

wherein said content cannot be copied onto another memory card and played back on a different portable device without knowledge of said player key, a new player ID, and said encryption algorithm.